大規模な制約なし最小化問題に対するコーダル部分グラフを用いたスパース準ニュートン法 (数値最適化の理論と実際)

AUTHOR(S):

黒川, 典俊; 山下, 信雄

CITATION:

黒川, 典俊 ...[et al]. 大規模な制約なし最小化問題に対するコーダル部分グラフを用いたスパース準ニュートン法 (数値最適化の理論と実際). 数理解析研究所講究録 2008, 1584: 72-83

ISSUE DATE:

2008-02

URL:

http://hdl.handle.net/2433/81499

数理解析研究所講究録
第1584巻 2008年 72-83

大規模な制約なし最小化問題に対するコーダル部分グラフを用いたスパース準ニュートン法

京都大学・情報学研究科　黒川 典俊　(Noritoshi Kurokawa)
山下 信雄　(Nobuo Yamashita)
Graduate School of Informatics,
Kyoto University

1 はじめに

本稿では，以下の制約なし最小化問題を考える．

$$\text{minimize} \quad f(x) \qquad \text{subject to} \quad x \in \mathbb{R}^n$$

ここで，$f:\mathbb{R}^n \to \mathbb{R}$ は2回連続的微分可能な関数とする．また，特に n が大きく，f のヘッセ行列 $\nabla^2 f(x)$ が疎，つまりほとんどの成分が0となる場合を考える．中小規模の制約なし最小化問題に対しては，準ニュートン法が有効な解法として広く使われている．準ニュートン法は，ヘッセ行列の近似行列を用いて点列を生成する反復法であり，実装が容易で収束の性質がよいという特徴を持つ．しかし，準ニュートン法の各反復で更新される近似ヘッセ行列は，一般的な更新公式（DFP公式やBFGS公式）[3]を用いた場合，密な行列となる．そのため，大規模な問題に対して適用するには何らかの工夫が必要である．

本稿では，$\nabla^2 f(x)$ の疎性を利用した準ニュートン更新の手法として近年提案された，行列補完を用いた準ニュートン法(Matrix Completion Quasi-Newton method, MCQN法) [5]を取り扱う．本稿を通して，$\nabla^2 f(x_k)$ の近似行列を B_k とし，その逆行列を H_k とする．また，$V := \{1,2,\cdots,n\}, E := \{(i,j) \mid [\nabla^2 f(x)]_{ij} \neq 0 \text{ for some } x \in \mathbb{R}^n\}$ とし，集合 E を $\nabla^2 f(x)$ の疎構造と呼ぶ．さらに，頂点集合を V，枝集合を $\overline{E} := E \setminus \{(i,i) \mid i=1,\cdots,n\}$ とする無向グラフ $G=(V,\overline{E})$ を $\nabla^2 f(x)$ の疎構造グラフと呼ぶ（今後，単に疎構造グラフと呼ぶ．）

$\nabla^2 f(x_k)$ の疎構造グラフがコーダルグラフであるとき，MCQN法で更新される行列 H_k は，疎な三角行列の積で表すことができる．しかし，一般には疎構造グラフはコーダルグラフとはならないため，[5]では疎構造グラフにいくつか枝を付け加えてコーダルグラフにし，得られたグラフ（コーダル拡張グラフ）の構造を用いて H_k を更新する手法が提案されている．

コーダル拡張グラフを用いて H_k を更新すると，MCQN法で生成される点列はある仮定の下で最適解に超一次収束するという長所をもつ．その一方で，問題によってはコーダル拡張グラフの枝数がもとの疎構造グラフの枝数と比べて大幅に増えるため，B_k の非ゼロ要素数が $\nabla^2 f(x_k)$ の非ゼロ要素数と比べて大幅に増えることがある．

本研究の目的は，その欠点を解消することである．本稿では，疎構造グラフからいくつか枝を削ってコーダルグラフにしたコーダル部分グラフを用いて，MCQN法の行列更新を行うことを考える．このとき，MCQN法の反復1回あたりの時間計算量と領域計算量を削減できることを示す．さらに数値実験の結果から，提案手法の有効性と今後の課題について考察する．

2 準備（コーダルグラフの基本的性質）

本節では，コーダルグラフに関する基本的な性質をまとめる（詳細は[1]を参照されたい）．本稿を通して，V を頂点の集合，$E \subseteq V \times V$ を枝の集合とし，$G=(V,E)$ で無向グラフを表すものとする．グラフにはループがない，すなわち，すべての $v \in V$ に対して $(v,v) \notin E$ と仮定する．

本稿で用いるグラフに関する用語を，以下で定義する．

定義 1 (基本的な用語)

- 2 つの頂点 $u, v \in V$ は $(u, v) \in E$ のとき隣接しているという.
- $v \in V$ に隣接している頂点の集合を $\mathrm{Adj}_G(v) = \{u \in V \mid (u, v) \in E\}$ で表す. 紛れのないときは, G を省略して, 単に $\mathrm{Adj}(v)$ とかく.
- $v \in V$ に接続している枝の数を v の次数といい, $\deg(v)$ で表す[*1].
- 相異なる 2 つの頂点がすべて隣接しているとき, グラフは完全であるという.
- 2 つのグラフ $G = (V, E)$ と $G' = (V', E')$ に対して, $V' \subseteq V$ かつ $E' \subseteq E$ が成り立つとき, G' を G の部分グラフという.
- V' をグラフ $G = (V, E)$ の頂点集合 V の部分集合とする. このとき, 頂点集合を V', 枝集合を $E' := E \cap (V' \times V')$ とするグラフ $G' = (V', E')$ を, (V' による) G の誘導部分グラフという[*2].
- C をグラフ $G = (V, E)$ の頂点集合 V の部分集合とする. このとき, C による $G = (V, E)$ の誘導部分グラフが完全であるならば[*3], その部分グラフを G のクリークという. クリーク中の任意の 2 頂点間には枝があることから, 本稿ではそれに属する頂点集合のみを明示して, クリーク C と表す.
- ある頂点 $v \in V$ に隣接する頂点の集合 $\mathrm{Adj}(v)$ が, グラフ $G = (V, E)$ 上でクリークを形成しているとする. このとき, その頂点 v を単体的頂点という.
- 他のクリークの真の部分グラフにならないクリークを, 極大クリークという. さらに, あるグラフの極大クリークの集合をそのグラフの極大クリーク族という.
- サイクル中の連続していない 2 つの頂点を結ぶ枝を弦 (コード) という.

定義 2 (コーダルグラフ) グラフ $G = (V, E)$ に含まれる長さ 4 以上のすべてのサイクルが弦をもつとき, G は コーダルグラフである, または単にコーダルであるという.

コーダルグラフの最も基本的な性質は, 次の 2 つである.

性質 1 コーダルグラフは, 単体的頂点をもつ.

性質 2 $G = (V, E)$ をコーダルグラフとし, $v_1 \in V$ をその単体的頂点とする. このとき, $V \setminus \{v_1\}$ により誘導される部分グラフもまたコーダルである.

性質 1,2 より, $V \setminus \{v_1\}$ により誘導される部分グラフも単体的頂点をもつ. それを v_2 とする. これを繰り返すことで, $\mathrm{Adj}(v_i) \cap \{v_{i+1}, v_{i+2}, \cdots, v_n\}$ がすべての $i = 1, 2, \cdots, n-1$ に対してクリークになるように, G の頂点に順序付け $(v_1, v_2, \cdots v_n)$ (ただし, $n = |V|$) を行うことができる. この順序を完全消去順序(**perfect elimination ordering**, 以後 **PEO** と表す) と呼ぶ. PEO の存在は, 次のようにコーダル性を特徴づけている.

性質 3 グラフ $G = (V, E)$ がコーダルであるための必要十分条件は, G が PEO をもつことである.

ただし, PEO は唯一ではないことに注意しておく.

さて, コーダルグラフの極大クリークは, PEO を用いて次のように簡単に列挙することができる. $G = (V, E)$ をコーダルグラフとし, $(v_1, v_2, \cdots, v_n)$ を G の PEO とする. v_1 は単体的頂点なので, v_1 を含む極大クリークは唯一であり, $\{v_1\} \cup \mathrm{Adj}(v_1)$ で与えられる. そして, v_1 を含まない極大クリークは, $\{v_2, v_3, \cdots, v_n\}$ から導かれた部分グラフの極大クリークである. したがって, コーダルグラフ $G = (V, E)$ の極大クリーク族

[*1] すなわち, $\deg(v) = |\mathrm{Adj}_G(v)|$ である.

[*2] G' は G の部分グラフになっていることに注意

[*3] すなわち, 異なる $i, j \in C$ のすべてのペアに対して $(i, j) \in E$

は，$\{v_i\} \cup (\mathrm{Adj}(v_i) \cap \{v_{i+1}, v_{i+2}, \cdots, v_n\}), i = 1, 2, \cdots, n$ の中で極大なものの集合として与えられる．よって，l を G の極大クリークの総数とすると，次の性質が成り立つ．

性質 4 $G = (V, E)$ をコーダルグラフとし，$(v_1, \cdots, v_n)$ を G の PEO とする．このとき，G の極大クリーク族 $\{C_r \mid r = 1, 2, \cdots, l\}$ は，次のように構成することができる：

$$C_r = \{v_i\} \cup (\mathrm{Adj}(v_i) \cap \{v_{i+1}, v_{i+2}, \cdots, v_n\}), \quad i = \min_{v_j \in C_r} j$$

性質 4 から，極大クリークの数 l の上界は n であることがわかる．

さらに，コーダルグラフの極大クリーク族は，次の条件を満たすように添え字をつけることができることが知られている．

性質 5 $G = (V, E)$ がコーダルグラフであるとき，$r = 1, 2, \cdots, l-1$ に対して，

$$\exists s \geq r+1 : C_r \cap (C_{r+1} \cup C_{r+2} \cup \cdots \cup C_l) \subseteq C_s \tag{1}$$

となる極大クリーク族 $\{C_r \mid r = 1, 2, \cdots, l\}$ が存在する．

性質 5 は **Running Intersection Property** （以降 **RIP** と表す）と呼ばれる．

3 MCQN 法

準ニュートン法は，目的関数のヘッセ行列（もしくはその逆行列）の近似行列を用いて点列を生成する反復法である．x_k を現在の反復点とし，H_k を目的関数のヘッセ行列 $\nabla^2 f(x_k)$ の近似逆行列とする．準ニュートン法では，まず探索方向を

$$p_k = -H_k \nabla f(x_k)$$

で定める．次に，p_k 方向に直線探索を行って，次回の反復点 x_{k+1} を

$$x_{k+1} = x_k + \alpha_k p_k$$

により定める．ただし，α_k は正の実数（ステップ幅）である．

行列 H_k は各反復において適宜更新する．代表的な更新公式として BFGS 公式：

$$(H_{k+1})_{ij} = (H_k)_{ij} + \rho s_i s_j - \frac{(H_k y_k)_i (s_k)_j + (s_k)_i (H_k y_k)_j}{s_k^T y_k} \quad \forall (i, j) \in V \times V \tag{2}$$

（ただし，$s_k = x_{k+1} - x_k$，$y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$，$\rho = \dfrac{1}{s_k^T y_k} + \dfrac{(y_k)^T H_k y_k}{(s_k^T y_k)^2}$）や DFP 公式などが知られている．BFGS 公式や DFP 公式で求まる H_{k+1} は，$s_k (s_k)^T$ などの影響で，$\nabla f(x)$ が疎であっても一般には密な行列になることに注意されたい．

MCQN 法の基本的な考え方は，$\nabla^2 f(x)$ の疎性を保存するため，H_k を次の要件を満たすように更新しようというものである．

- $(H_{k+1})_{ij}, \forall (i, j) \in F$ は BFGS 公式などで更新する．
- $(H_{k+1}^{-1})_{ij} = 0, \forall (i, j) \notin F$ を満たす．
- H_{k+1} は正定値行列．

ここで，集合 F は，$F \subseteq V \times V$ かつ $F \approx E$ を満たすように選ぶ．また，(a) $(i,i) \in F, i = 1,2,\cdots,n$ (b) $(i,j) \in F \Rightarrow (j,i) \in F$ と仮定する．集合 F はなるべく $F = E$ となるように選ぶのが望ましいが，F にはコーダルグラフに関する条件を課す必要があるので，必ずしも $F = E$ とできない．F の選び方については後述する．

さて，上の要件を満たす行列は，以下のようにして得られる．

Step 1: BFGS 公式 (2) などを用いて H_k から $(\overline{H}_{k+1})_{ij}, \forall (i,j) \in F$ の成分を計算しておく．

Step 2: $(\overline{H}_{k+1})_{ij}, (i,j) \in F$ を用いた最適化問題：

$$\begin{array}{ll} \min_H & \psi(H_k^{-\frac{1}{2}} H H_k^{-\frac{1}{2}}) \\ \text{subject to} & H_{ij} = (\overline{H}_{k+1})_{ij} \quad \forall (i,j) \in F \\ & (H^{-1})_{ij} = 0 \quad \forall (i,j) \notin F \\ & H = H^T,\ H \succeq 0 \end{array} \tag{3}$$

の最適解を H_{k+1} とする[*4]．

ここで，$\psi : \mathbb{R}^{n \times n} \to \mathbb{R}$ は $\psi(A) = \mathrm{trace}(A) - \ln \det(A)$ で定義される狭義凸関数である．問題 (3) は，DFP 公式を与える最適化問題 [2] に，「ヘッセ行列の疎性の条件 $((H^{-1})_{ij} = 0,\ \forall (i,j) \in F)$」を付け加えた問題の近似問題である [5]．

問題 (3) は次の問題と等価であることが知られている．

$$\begin{array}{ll} \max_H & \det H \\ \text{subject to} & H_{ij} = (\overline{H}_{k+1})_{ij} \quad \forall (i,j) \in F \\ & (H^{-1})_{ij} = 0 \quad \forall (i,j) \notin F \\ & H = H^T,\ H \succeq 0 \end{array} \tag{4}$$

問題 (4) は半正定値行列補完問題であり一般には難しいが，F が次の条件を満たすとき，その解を陽に表すことができる．

【条件】頂点集合を V，枝集合を $\overline{F} := F \setminus \{(i,i) \mid i = 1,\cdots,n\}$ とする無向グラフ $G' = (V, \overline{F})$ がコーダルグラフとなる．

実際，F が上の条件を満たすとき，問題 (4) の解 H^* は（$(\overline{H}_{k+1})_{ij}, (i,j) \in F$ のみから計算できる）疎行列の積として，次のように表される [5, 6]．（この結果は，正定値行列補完の理論を用いて導き出されている．）

定理 3 グラフ $G' = (V, \overline{F})$ はコーダルグラフであるとし，$\{C_r\}, r = 1,2,\cdots,l$ をグラフ $G' = (V, \overline{F})$ の RIP を満たす極大クリーク族（l は極大クリークの総数）とする．

集合族 $\{S_r\}, \{U_r\}$ を

$$S_r := C_r \setminus (C_{r+1} \cup C_{r+2} \cup \cdots \cup C_l), \quad r = 1,\cdots,l \tag{5}$$

$$U_r := C_r \cap (C_{r+1} \cup C_{r+2} \cup \cdots \cup C_l), \quad r = 1,\cdots,l \tag{6}$$

で定義する．（ここで，$V = \bigcup_{r=1}^{l} S_r$ かつ $S_i \cap S_j = \emptyset$ であることに注意しておく．）さらに，$S_1, S_2, \cdots, S_l$ の順にそれぞれの要素を取り出して並べたときに $1,2,\cdots,n$ となるようにグラフ G' の頂点の番号をつけかえる．その置換行列を P としたとき，問題 (4) の解 H^* は，次のように疎なブロック行列の積で表される：

$$H^* = P^T L_1^T L_2^T \cdots L_{l-1}^T D L_{i-1} \cdots L_2 L_1 P \tag{7}$$

[*4] $H \succeq 0$ は，H が半正定値であることを表す．同様に，$H \succ 0$ は，H が正定値であることを表す．

ここで、$r=1,2,\cdots,l-1$ に対して

$$[L_r]_{ij}=\begin{cases}1 & i=j\\ \left[\left(\overline{H}_{U_rU_r}\right)^{-1}\overline{H}_{U_rS_r}\right]_{ij} & (i,j)\in U_r\times S_r\\ 0 & \text{otherwise}\end{cases} \tag{8}$$

であり（表記の簡単のため、$\overline{H}_{k+1}$ を $\overline{H}$ と略記した．また、$n\times n$ 行列 A と $S,U\subseteq V$ に対し，A_{SU} は $A_{ij},(i,j)\in S\times U$ を要素とする $|S|\times|U|$ 行列（A の小行列）を表す），

$$D=\begin{pmatrix}D_{S_1S_1} & & & \\ & D_{S_2S_2} & & \\ & & \ddots & \\ & & & D_{S_lS_l}\end{pmatrix} \tag{9}$$

である．ただし，

$$D_{S_rS_r}=\begin{cases}\overline{H}_{S_rS_r}-\overline{H}_{S_rU_r}\left(\overline{H}_{U_rU_r}\right)^{-1}\overline{H}_{U_rS_r} & r\le l-1\\ \overline{H}_{S_rS_r} & r=l\end{cases} \tag{10}$$

である．

この定理は，$\nabla^2 f(x_k)$ の疎構造グラフがコーダルグラフであるとき，MCQN 法で更新される行列 H_k が，疎行列の積で表せることを保証している．また，問題 (4) の形から，$s_k^Ty_k>0$ かつ $H_k\succ 0$ が満たされていれば，$H_{k+1}\succ 0$ が従うことにも注意する[*5]．

しかし，一般には疎構造グラフ $G=(V,\overline{E})$ はコーダルグラフとはならないため，疎構造グラフに何らかの操作を加えてコーダルグラフにし（本稿では，この作業を「疎構造グラフのコーダル化」と呼ぶことにする），得られたグラフ $G'=(V,\overline{F})$ を用いて行列を更新することが考えられる．その手法は次節で議論することにし，MCQN 法のアルゴリズムの概要を以下に記す．

MCQN 法のアルゴリズム

$V=\{1,2,\cdots,n\},E=\{(i,j)\,|\,\text{ある } x\in\mathbb{R}^n\text{に対して } [\nabla^2 f(x)]_{ij}\neq 0\ \},\overline{E}=E\setminus\{(i,i)\,|\,i=1,\cdots,n\}$ とする．

Step 0 : （疎構造グラフのコーダル化と初期化）

(0-1) $G=(V,\overline{E})$ をコーダル化したグラフ $G'=(V,\overline{F})$ を求める．

(0-2) $G'=(V,\overline{F})$ の RIP を満たす極大クリーク族 $\{C_r\,|\,r=1,2,\cdots,l\}$ を求める
（G' の RIP を満たす極大クリーク族を求めるアルゴリズムは，付録 A.2 を参照）．
式 (5),(6) で定義される集合族 S_r,U_r を計算する．$S_1,S_2,\cdots,S_l$ の順にそれぞれの要素を取り出して並べたときに $1,2,\cdots,n$ となるように G' の頂点の番号をつけかえる．その置換行列 P を求める．

(0-3) $F=\overline{F}\cup\{(i,i)\,|\,i=1,\cdots,n\}$ とする．
初期点 $x_0\in\mathbb{R}^n$ と，$(H_0^{-1})_{ij}=0,\ \forall(i,j)\notin F$ を満たす正定値対称行列 H_0 を選ぶ．
$k:=0$ とする．

Step 1 : （探索方向の決定）$p_k=-H_k\nabla f(x_k)$ とする．

[*5] $s_k^Ty_k>0$ かつ $H_k\succ 0$ が満たされていれば $\overline{H}_{k+1}\succ 0$ が従うので，$\det\overline{H}_{k+1}\succ 0$. $\overline{H}_{k+1}$ は問題 (4) の実行可能解だから，問題 (4) の最適解 H^* に対し，$\det H^*\ge\det\overline{H}_{k+1}$ が成立することよりわかる．

Step 2 : (次回の反復点の決定)

(2-1) p_k 方向に直線探索を行い，適当な直線探索の基準（例えば，Wolfe の条件 [3]）を用いてステップ幅 α_k を決定する．

(2-2) $x_{k+1} = x_k + \alpha_k p_k$ とおく．

Step 3 : 停止条件が満たされていれば，x_{k+1} を解とみなして停止する．さもなければ Step 4 へいく．

Step 4 : (近似行列の更新)

(4-1) $s_k = x_{k+1} - x_k, y_k = \nabla^2 f(x_{k+1}) - \nabla^2 f(x_k)$ を計算する．

(4-2) 既存の準ニュートン法の更新公式 (BFGS 公式, DFP 公式など) により，$\left(\overline{H}_{k+1}\right)_{ij}, \forall (i,j) \in F$ を求める．(例えば，BFGS 公式を用いるなら，式 (2) で計算できる)

(4-3) 式 (7)〜(10) により，$H_{ij} = \left(\overline{H}_{k+1}\right)_{ij}, \forall (i,j) \in F$ を満たす正定値行列 H を求め，それを H_{k+1} とする．

$k := k+1$ として，Step1 へ戻る．

Step 4-3 は，実はダミーステップである．まず，2 節で述べたアルゴリズムより，H_k が必要になるのは，Step 2 で $p_k = -H_k \nabla f(x_k)$ を計算するときと，Step 4-2 で $\left(\overline{H}_{k+1}\right)_{ij}$ $\forall (i,j) \in F$ を求める際に $H_k y_k$ を計算するときの 2 回だけである．これらはいずれも行列 H_k とあるベクトルの積演算になっている．さらに，$(H_k)_{ij}, \forall (i,j) \notin F$ の成分は $\left(\overline{H}_{k+1}\right)_{ij}, \forall (i,j) \in F$ のみから式 (8)〜(10) を用いて計算できることに注意すると，MCQN 法の実装の際には，H_k を陽に計算する必要はなく，行列 H_k とあるベクトル $d \in \mathbb{R}^n$ の積 $H_k d$ を計算するアルゴリズムを実装すればよいことがわかる．その計算は，次のように実行することができる．

まず，任意のベクトル $d \in \mathbb{R}^n$ に対し，$H_k d$ の計算は，式 (7) を用いて，

$$H_k d = P^T L_1^T L_2^T \cdots L_{l-1}^T D L_{l-1} \cdots L_2 L_1 P d$$

で計算できることに注意する．これを右から順番に，

$$\begin{aligned} q_1 &= L_1 P q_0 \\ q_2 &= L_2 q_1 \\ &\vdots \\ q_{l-1} &= L_{l-1} q_{l-2} \\ q_l &= D q_{l-1} \\ q_{l+1} &= [L_{l-1}]^T q_l \\ &\vdots \\ q_{2l-1} &= P^T L_1^T q_{2l-2} \end{aligned}$$

と計算すれば，$H_k d = q_{2l-1}$ と求められる．

4 集合 F の選び方と計算量

4.1 集合 F の選び方

$\nabla^2 f(x)$ の疎構造グラフ $G = (V, \overline{E})$ は一般にコーダルグラフとは限らないので，G の枝集合に操作を加えてコーダルグラフにしたグラフ $G' = (V, \overline{F})$ を求める必要がある．

本稿では領域計算量を抑えるため，集合 F を (a) $F \subseteq E$ かつ (b) $G' = (V, \overline{F})$ がコーダルグラフ，となるように選ぶ手法を提案する．この条件を満たす集合 F を求めることは，疎構造グラフのコーダル部分グラフを求めることと等価である．

このように F を選ぶと，B_k の非ゼロ要素数は $\nabla^2 f(x)$ の非ゼロ要素数以下に抑えられ，陽に計算すべき H_k の成分も少なくてすむ．しかし，$F \subsetneq E$ であるとき，ヘッセ行列の情報が省かれてしまうため，MCQN 法が持つ高速性が失われてしまう可能性がある．したがって，もとの疎構造グラフの枝数最大のコーダル部分グラフを用いることが望ましい．一般のグラフに含まれる枝数最大のコーダル部分グラフを見つける問題は NP 完全であるため，それを近似的に解く実用的なヒューリスティックアルゴリズムがいくつか提案されている．その中でも，Xue のアルゴリズム [4] は比較的枝数の多いコーダル部分グラフを得られることで知られている（アルゴリズムは付録 A.1 を参照）．

4.2 MCQN 法の計算量

まず，MCQN 法の反復 1 回あたりの時間計算量を評価する．与えられた $d \in \mathbb{R}^n$ に対し，$H_k d$ を前節で述べた手順で計算するとする．あるベクトル $w \in \mathbb{R}^n$ に対して，各 $L_r w$ の時間計算量は $O(|U_r||S_r|)$ であり，Dw の時間計算量は $O\left(\sum_{r=1}^{l} |S_r|^2\right)$ であることから[*6]，$H_k d$ の時間計算量は $O\left(\sum_{r=1}^{l}(|U_r||S_r| + |S_r|^2)\right) = O\left(\sum_{r=1}^{l} |C_r|^2\right)$ である．

以上のことに注意すると，Step 2 における探索方向の時間計算量は，($\nabla f(x_k)$ が与えられているとすると) $O\left(\sum_{r=1}^{l} |C_r|^2\right)$ である．また，Step 3 の時間計算量は，Step 2 と比べればほとんど無視できる．

Step 4 における行列更新に必要な反復 1 回あたりの時間計算量を評価する．まず，$[\overline{H}_{k+1}]_{ij}, \forall (i,j) \in F$ は従来の準ニュートン法の更新公式で求められる．例えば BFGS 公式で更新されるとすると，

$$\left(\overline{H}_{k+1}\right)_{ij} = (H_k)_{ij} + \rho s_i s_j - \frac{(H_k y_k)_i (s_k)_j + (s_k)_i (H_k y_k)_j}{s_k^T y_k} \quad \forall (i,j) \in F$$

で計算される．よって，$H_k y_k$ が計算済みならば，$[\overline{H}_{k+1}]_{ij}, \forall (i,j) \in F$ の計算は $O(|F|)$ でできる．$H_k y_k$ の時間計算量は $O\left(\sum_{r=1}^{l}(|U_r||S_r| + |S_r|^2)\right)$ なので，$[\overline{H}_{k+1}]_{ij}, \forall (i,j) \in F$ の時間計算量は

$$O\left(|F| + \sum_{r=1}^{l}(|U_r||S_r| + |S_r|^2)\right) = O\left(\sum_{r=1}^{l} |C_r|^2\right)$$

である．

次に，式 (8)〜(10) で与えた $L_r, r = 1, 2, \cdots, l$ と D の時間計算量を評価する．以下でも $\overline{H}_{k+1}$ を $\overline{H}$ と略記する．まず，$\overline{H}_{U_r U_r}, \overline{H}_{S_r U_r}, \overline{H}_{S_r S_r}, r = 1, 2, \cdots, l$ の時間計算量はそれぞれ $O(|U_r|^2), O(|U_r| \times |S_r|), O(|S_r|^2)$ でなる．また，$(\overline{H}_{U_r U_r})^{-1}$ の時間計算量は $O(|U_r|^3)$ となる．したがって，D と $L_r, r = 1, 2, \cdots, l$ の時間計算量は大まかに $O\left(\sum_{r=1}^{l} |C_r|^3\right)$ となる．なお，$[(H_k)_{U_r U_r}]^{-1}$ をすべての $r = 1, 2, \cdots, l$ について蓄えておくと，Sherman-Morrison の公式 [3] を使うことによって，時間計算量は $O\left(\sum_{r=1}^{l} |C_r|^2\right)$ に減らすことができる．

[*6] $L_r, r = 1, 2, \cdots, l-1$ は，

$$L_r = I + M_r, \quad M_r = \begin{cases} [L_r]_{ij} & (i,j) \in U_r \times S_r \\ 0 & \text{otherwise} \end{cases}$$

とかけるので，$L_r w = (I + M_r)w = w + M_r w$ となる．M_r には非ゼロ要素が $|U_r| \times |S_r|$ 個しかないことを考えると，$M_r w$ の計算は，$O(|U_r||S_r|)$ でできる．ゆえに，$L_r w$ の時間計算量は $O(|U_r||S_r|)$ である．

Dw の計算は，

$$Dw = \begin{pmatrix} D_{S_1 S_1} w_{S_1} \\ D_{S_2 S_2} w_{S_2} \\ \vdots \\ D_{S_l S_l} w_{S_l} \end{pmatrix}$$

であることを考えると，$O\left(\sum_{r=1}^{l} |S_r|^2\right)$ でできる．ここで，w_{S_r} は $w_i, i \in S_r$ を要素とする $|S_r|$ 次元ベクトルである．

領域計算量は，$(H_k)_{ij}, \forall(i,j) \in F$ のみをメモリに蓄えたとき $O(|F|)$，$(H_k)_{ij}, \forall(i,j) \in F$ および $[(H_k)_{U_rU_r}]^{-1}, r = 1,2,\cdots,l-1$ をメモリに蓄えたとき $O\left(|F| + \sum_{r=1}^{l} |U_r|^2\right)$ である．

以上の評価から，MCQN 法の計算量は

- $(H_k)_{ij}, \forall(i,j) \in F$ のみをメモリに蓄えたとき
  - 領域計算量は $O(|F|)$
  - 反復 1 回あたりの時間計算量は $O\left(\sum_{r=1}^{l} |C_r|^3\right)$
- $(H_k)_{ij}, \forall(i,j) \in F$ および $[(H_k)_{U_rU_r}]^{-1}, r = 1,2,\cdots,l-1$ をメモリに蓄えたとき
  - 領域計算量は $O\left(|F| + \sum_{r=1}^{l} |U_r|^2\right) \leq O\left(\sum_{r=1}^{l} |C_r|^2\right)$
  - 反復 1 回あたりの時間計算量は $O\left(\sum_{r=1}^{l} |C_r|^2\right)$

である．

ヘッセ行列が疎であるとき，一般に $|C_r| \ll n$ である．2 節の性質 4 から $l \leq n$ なので，$\sum_{r=1}^{l} |C_r|^2 \ll n^2$ が成り立つ．特にヘッセ行列が三重対角行列であるときは，行列のサイズを n とすると，$l = n$ かつ $r = 1,2,\cdots,n$ に対して $|C_r| = 2$ が成り立つ．このとき，$(H_k)_{ij}, \forall(i,j) \in F$，およびすべての $r = 1,2,\cdots,l$ に対して $[(H_k)_{U_rU_r}]^{-1}$ をメモリに蓄えたとしても，領域計算量と反復 1 回あたりの時間計算量はともに $O(n)$ となる．

上の議論から，MCQN 法の計算量に最も影響を及ぼすのは，疎構造グラフ G をコーダル化したグラフ (G') に含まれる極大クリークの大きさ[*7]であることがわかる．コーダル縮小に含まれるクリークの大きさはコーダル拡張に含まれるクリークの大きさよりも小さいことから，コーダル縮小を用いた MCQN 法の反復 1 回あたりの計算量は時間計算量，領域計算量ともにコーダル拡張グラフを用いた MCQN 法よりも減らすことができる．

なお，疎構造グラフ G のコーダル縮小を求める操作は，MCQN 法の前処理として行う．その時間計算量は，Xue のアルゴリズム [4] を用いた場合，$O\left(|V| + \sum \deg(v) + \sum \deg^2(v)\right) \sim O(\Delta \times |\overline{F}|)$ である．（ただし，$\Delta = \max\{\deg(v) \mid v \in V\}$ は，グラフ $G = (V, \overline{F})$ の頂点の最大次数である．）ヘッセ行列が疎であるとき，$|\overline{F}|$ は小さく，$\deg(v) \ll n$ となるので，この時間計算量は MCQN 法の行列更新にかかる時間計算量よりも少ない．

5 数値実験

本節では，制約なしの凸 2 次計画問題

$$\text{minimize} \quad f(x) = \frac{1}{2}x^T Ax + b^T x$$

を MCQN 法で解いた際の数値実験の結果を報告する．実験は CPU が 3.4GHz の Pentium4，メモリが 3.5GB の計算機上で行い，アルゴリズムは MATLAB7.0 を用いて実装した．

本実験では，コーダル拡張を用いた場合（以下 Ext-MCQN 法）とコーダル縮小を用いた場合（以下 Del-MCQN 法）とで，

1. 最適解が得られるまでの反復回数
2. 反復 1 回あたりの計算コスト

を比較した．

[*7] 極大クリークを $C_r, r = 1,2,\cdots,l$ とすると，$|C_r|$ のこと．

Del-MCQN 法，Ext-MCQN 法の反復 1 回あたりの計算コストとしては，それぞれ次で定義される値を用いた[*8]．

$$\mathrm{Del\_cost} = \sum_r |C'_r|^2$$
$$\mathrm{Ext\_cost} = \sum_r |C''_r|^2$$

ただし，$\{C'_r\}, \{C''_r\}$ はそれぞれ G のコーダル縮小 G' の極大クリーク族，G のコーダル拡張 G'' の極大クリーク族である．

本実験では，行列 A のサイズ n を $n = 1000$ と固定した．行列 A は，MATLAB の関数 sprandsym を用いて，以下の式によって求めた．

$$A = \mathrm{sprandsym}(n, nz, rc, option)$$

sprandsym は，条件数がおよそ $1/rc$ となり，非ゼロ要素率 (= 非ゼロ要素の数 $/n^2$) がおよそ nz となるような，$n \times n$ の正定値対称行列をランダムに生成する関数である．$option$ は，行列の生成方法を指定するための引数で，今回は $option = 2$ と指定した[*9]．なお，ベクトル b は $[0,1]^n$ からランダムに選んだ．すべての実験において，初期点 x_0 は $x_0 := (100, \cdots, 100)^T \in \mathbb{R}^n$ とし，初期行列 H_0 は $H_0 := I$ (n 次単位行列) とした．

また，$\overline{H}_{ij}, \forall (i,j) \in F$ は BFGS 公式 (2) を用いて求めた．H_{k+1} の正定値性を保証するため，$s_k^T y_k \leq 2.2 \times 10^{-16}$ のときは $H_{k+1} = H_k$ とした．アルゴリズムの終了条件には，

$$\|\nabla f(x_k)\| < 10^{-5}$$

を用いた．

行列 A の条件数を変化させたときの反復回数の比較を表 1 にまとめる（この結果は，行列 A の非ゼロ要素率を 1% に固定した場合である．）．また，反復 1 回あたりの計算コストの比較を表 2 にまとめる．（なお，行列 A の条件数は Del_cost, Ext_cost の値に影響しないので，行列 A を生成する関数 sprandsym において，条件数（の逆数）を指定するパラメータ rc を 0.01 に固定した．）

表 1　反復回数の比較（A の非零要素率：1%）

	反復回数	
条件数	縮小	拡張
2.2	42	30
9.5	62	45
18.5	73	56
97.0	288	141
197.9	519	182
1307.6	2312	334

表 2　計算コストの比較

	計算コスト	
非ゼロ要素率	縮小	拡張
0.2%	2,682	2,913
0.4%	3,975	168,242
0.6%	4,277	744,299
0.8%	4,697	1,323,809
1.0%	5,115	1,661,678

この結果は次のように整理できる．

[*8] 前節で述べたとおり，Del-MCQN 法および Ext-MCQN 法の反復 1 回あたりの時間計算量は，それぞれ $O(\sum_r |C'_r|^2), O(\sum_r |C''_r|^2)$ である（$(H_k)_{ij}, \forall (i,j) \in F$，および $[(H_k)_{U_r U_r}]^{-1}, \forall r = 1, 2, \cdots, l$ をメモリに蓄えたとき）．Del-MCQN 法，Ext-MCQN 法の計算スキームは，前処理の段階でそれぞれコーダル縮小，コーダル拡張を求めるところを除いて同じであることに注意すれば，この値を用いて計算コストを比較することには意味がある．

[*9] $option = 2$ としたとき，関数 sprandsym で生成される行列の条件数は正確には $1/rc$ とならない．したがって，表 1 では生成された行列の条件数を掲載した．

	コーダル縮小	コーダル拡張
反復回数	A の条件数が大きくなると， 大幅に増える．	A の条件数が大きくなっても， コーダル縮小のときほどは増えない．
反復1回あたりの 計算コスト	A の非ゼロ要素率が大きくなっても それほど増加しない．	A の非ゼロ要素率が大きくなると， 急激に増加する．

したがって，行列 A の条件数がそれほど大きくない場合には，Del-MCQN 法は有効であるといえる．

6 まとめと今後の課題

本稿では，MCQN 法の概要について述べたあと，MCQN 法で必要となる疎構造グラフ G をコーダル化したグラフとして，G のコーダル部分グラフを利用する手法を提案した．理論的には（コーダル拡張グラフを用いた場合と比べて）MCQN 法の反復1回あたりの時間計算量と領域計算量を削減できることを示した．

また，疎構造グラフのコーダル部分グラフを求める発見的手法として知られる Xue のアルゴリズムを用いて数値実験を行った．その結果から，コーダル縮小を用いた MCQN 法について

- 行列の条件数を大きくしたり，非ゼロ要素率を大きくすると，反復回数は増加する
- 反復回数の観点からはコーダル拡張を用いた方法よりも劣っているが，反復1回あたりの計算にかかるコストの観点からは優れている．したがって，条件数がそれほど悪くない問題ではコーダル縮小を用いた MCQN 法のほうが全体の計算量も小さいといえる

ことがわかった．

まだ数多くの研究課題が残されている．以下にその課題をまとめる．

- 疎構造グラフの枝数に最も近いコーダルグラフの利用
  集合 F の選び方として，
  - コーダル拡張グラフ（疎構造グラフにいくつか枝を付け加えてコーダルグラフにしたもの）の構造を用いて H_k を更新する [5]
  - コーダル縮小グラフ（疎構造グラフからいくつか枝を削ってコーダルグラフにしたもの）の構造を用いて H_k を更新する（本稿）

  を述べたが，今後は，もとの疎構造グラフにいくつか枝をつけくわえたり削ったりしてコーダルグラフにしたもの[*10]を用いて MCQN 法を適用することも考えられる．疎構造グラフに付け加えたり削ったりする枝の数が少なければ，ヘッセ行列の疎構造をよりよく保存できると考えられる．
- 収束率の解明
  コーダル拡張を用いた MCQN 法により生成される点列は，適当な仮定のもとで最適解に超一次収束することが示されているが，コーダル縮小を用いた手法の収束率についてはまだ解明されていない．ある条件の下で1次収束性が示されるかどうかは今後の課題である．
- 目的関数のヘッセ行列の疎構造をいかに推定するか
  目的関数のヘッセ行列の疎構造が分からなければ，MCQN 法の手法は利用しにくい．そのため，ヘッセ行列のおおよその疎構造を推定するためのアルゴリズムの開発が望まれる．
- 一般の非線形計画問題に対する実験
  CUTEr のテスト問題に収録されている問題を用いていくつか予備実験を行ったが，ヘッセ行列の疎構造グラフがもとからコーダルグラフになっていたため，コーダル拡張を用いた手法と縮小を用いた手

[*10] 一般に，このようなコーダルグラフを見つける問題は chordal editing 問題として知られている．

法のパフォーマンスの差を比較することができなかった．今後は他の手法（記憶制限付き準ニュートン法など）との比較を行う必要がある．

参考文献

[1] J. R. S. Blair and B. W. Peyton: *An introduction to chordal graphs and clique trees*, in Graph Theory and Sparse Matrix Computation, A. George, J. R. Gilbert and J. W. H. Liu, eds., pp. 1–29, Springer-Verlag, New York (1993).

[2] R. Fletcher: *A new variational result for quasi-Newton formulae*, SIAM journal on Optimization, Vol. 1, No. 1, pp. 18–21 (1991).

[3] J. Nocedal and S. J. Wright: *Numerical Optimization*, Springer-Verlag, New York (1999).

[4] J. Xue: *Edge-maximal triangulate subgraphs and heuristics for the maximum clique problem*, Networks, Vol. 24, pp. 109–120 (1994).

[5] N. Yamashita: *Sparse quasi-newton updates with positive definite matrix completion*, to appear in Mathematical Programming.

[6] 黒川 典俊: 大規模な制約なし最小化問題に対するコーダル部分グラフを用いたスパース準ニュートン法, 京都大学工学部情報学科数理工学コース 特別研究報告書 (2007).

付録

付録 A　コーダルグラフに関するアルゴリズム

A.1　コーダル部分グラフ（コーダル縮小グラフ）を求めるアルゴリズム

与えられた任意のグラフに含まれる枝数最大のコーダル部分グラフを求める問題は NP-完全であり一般には難しい．そこで，それを近似的に解くヒューリスティックアルゴリズムがいくつか提案されている．ここでは，Xue のアルゴリズム [4] を紹介する．このアルゴリズムは，一般のグラフに対する最大クリーク問題への応用を念頭に考えられたアルゴリズムである．表記の簡単のため，$\mathrm{Suc}_G(v_i) := \{v_j \,|\, j > i \,\mathrm{and}\, v_j \in \mathrm{Adj}_G(v_i)\}$ とする．

Edge-maximal chordal subgraph (Xue [4])

入力： グラフ $G = (V, E)$

出力： グラフ G のコーダル部分グラフ（コーダル縮小）$G' = (V, E')$

Step 0： $k := n$ とし，$V^k := \emptyset$，$E^k := \emptyset$，$peo := \emptyset$，$U := V$ とする．また，$\forall v \in V$ に対し，$t(v) = \emptyset$，$s(v) = 0$ とする．

Step 1： $k = 1$ なら終了．$G^k := (V^k, E^k)$ は $peo = (v_1, \cdots, v_n)$ を PEO にもつ枝数最大のコーダル部分グラフである．

Step 2： $s(v) = \max\{s(u) \,|\, u \in U\}$ を満たす $v \in U$ を 1 つ選ぶ．$V^{k-1} := V^k \cup \{v\}$，$peo := (v, peo)$，$U := U \setminus \{v\}$ とする．$E^{k-1} := E^k \cup \{(v, u) \,|\, u = t(v) \,\mathrm{or}\, u \in \mathrm{Adj}_G(v) \cap \mathrm{Suc}_{G^k}(t(v))\}$ とする．頂点 v のラベルを v_k とする．

Step 3： $\forall u \in \mathrm{Adj}_G(v) \cap U$ に対し，$r_u := 1 + |\mathrm{Suc}_{G^k}(v) \cap \mathrm{Adj}_G(u)|$ とする．
もし，$r_u \geq s(u)$ なら，$t(u) := v$，$s(u) := r_u$ とする．$k := k - 1$ とし，Step 1 へ戻る．

注意 1 Step 2 において $\max\{s(u) \mid u \in U\}$ を満たす頂点が複数あるとき，Xue はグラフ G において次数が最大となるものを選ぶことを推奨している．

Xue のアルゴリズムで生成される G のコーダル縮小については，次の性質が知られている．

定理 4 ([4],Theorem 3.1) $G = (V, E)$ に対して Xue のアルゴリズムを適用して得られたコーダル縮小を $G' = (V, E')$ とする．G' は $peo = (v_1, \cdots, v_k)$ を PEO にもつ G のコーダル部分グラフの中で枝数最大のものである．

なお，Xue のアルゴリズムの計算量は，$O(|V| + \sum \deg(v) + \sum \deg^2(v)) \sim O(\Delta \times |E|)$ である [4]．ただし，$\Delta = \max\{\deg(v) \mid v \in V\}$ である．

A.2 RIP を満たす極大クリーク族とクリーク木を求めるアルゴリズム

コーダルグラフとその PEO から，RIP を満たす極大クリーク族とクリーク木は以下のアルゴリズムによって求めることができる [1]．

PEO から RIP を満たす極大クリーク族とクリーク木を求めるアルゴリズム

Step 1 : $r := 1, C_1 := \{v_n\}$, 極大クリーク族 $\mathcal{K}_n := \{C_1\}, \mathrm{parent}(C_1) := \emptyset, i := n-1$ とする．
Step 2 : $i = 0$ ならば終了．さもなければ $A_i := \mathrm{Adj}(v_i) \cap \{v_{i+1}, \cdots, v_n\}$ とする．
　　$C_q \supseteq A_i$ となるような $C_q \in \mathcal{K}_{i+1}$ を求める．
Step 3 : $C_q = A_i$ ならば，$C_q := C_q \cup \{v_i\}$ とする．
　　さもなければ，$r := r+1, C_r := A_i \cup \{v_i\}, \mathrm{parent}(C_r) := C_q, \mathcal{K}_i := \mathcal{K}_{i+1} \cup \{C_r\}$ とする．
Step 4 : $i := i-1$ とし，Step 2 へ．

このアルゴリズムでは，極大クリーク族 $\{C_r\}$ を求めると同時にクリーク木を求めている．クリーク木は各頂点（クリーク）の親頂点（クリーク）を表す関数 parent によって表現されている．